A Comparison of Point Counting methods for Hyperelliptic Curves over Prime Fields and Fields of Characteristic 2

Colm Ó hÉigeartaigh [1,2]

School of Computing
Dublin City University
Dublin, Ireland

Abstract

Computing the order of the Jacobian of a hyperelliptic curve remains a hard problem. It is usually essential to calculate the order of the Jacobian to prevent certain sub-exponential attacks on the cryptosystem. This paper reports on the viability of implementations of various point-counting techniques. We also report on the scalability of the algorithms as the fields grow larger.

Key words: Hyperelliptic Curve Cryptography, lattices, point counting, Jacobian

1 Introduction

In the late 1980s, Elliptic Curve Cryptography emerged as an alternative to finite-field based Public Key Cryptosystems, such as RSA[2] or El-Gamal[3]. For example, the El-Gamal cryptosystem (as originally described) relies on the Discrete Logarithm Problem over finite fields. A number of attacks called Index Calculus Attacks[1] offered subexponential attacks against the Discrete Logarithm Problem over finite fields, meaning that the field sizes had to be increased all the time to maintain security levels. This led to slower encryption/decryption times, and increased key-sizes.

The points on an elliptic curve form a group together with the ”point addition” group law. The Discrete Logarithm Problem is then defined using this group. The Index Calculus Attacks mentioned previously do not apply to Elliptic Curves, due to the difficulty in obtaining a factor base. This means that the group law of Elliptic

[1] This author wishes to thank Enterprise Ireland for their support with this research under grant IF/2002/0312/N

[2] Email: `coheigeartaigh@computing.dcu.ie`

Curve Cryptography is substantially faster than that of RSA[2], as the defining field is a lot smaller. For example, it is reckoned that a 1024-bit RSA key is equivalent in security terms to a 160-bit key for Elliptic Curves [4].

In recent years, people have begun to look at a richer source of groups than Elliptic Curves, namely the Jacobian Variety of Hyperelliptic Curves. Researchers began looking at Hyperelliptic Curve Cryptography [5] because the order of the Jacobian of a Hyperelliptic Curve of genus g over a field with q elements is $\simeq q^g$. This means that if you have an Elliptic Curve with a field size of $q \simeq 2^{160}$, then a Hyperelliptic Curve of genus 2 and 3, can have field sizes 2^{80} and 2^{53} respectively. Smaller field sizes, lead to smaller power consumption, which is an advantage when implementing Hyperelliptic Curve Cryptography on embedded devices, such as smartcards[6].

Determining the group order is important for the security of the cryptosystem. For example the order of the Jacobian must be divisible by a large prime to avoid Shanks' Baby-Step Giant-Step method and the Pohlig-Hellman method[7]. Computing the order of the Jacobian of a hyperelliptic curve remains a hard problem. Gaudry presented a method [8] to determine the group order for generic curves of genus 2, which has a running time of about a week. This might be too slow for researchers attempting to determine the security of a particular hyperelliptic curve cryptosystem.

In this paper, we have implemented various point counting techniques for specific types of curves, or over specific fields. We have implemented two algorithms over fields of characteristic 2, and four algorithms over prime fields. We detail these algorithms, the running times of each algorithm, and comment upon their viability. We also comment on the scalability of the algorithms as field sizes increase.

We focus on point counting algorithms for genus 2 curves in this paper, as they seem to be less vulnerable to attack than genus 3 or 4 curves.

2 Brief Facts on Hyperelliptic Curves

A non-singular (imaginary quadratic) hyperelliptic curve C of genus g over a field $\mathbb{F}_k$ is defined by an equation of the form: $C : v^2 + h(u)v = f(u)$, where $h, f \in k[u]$, f is monic, and the degree of $f = 2g + 1$, deg $h \leq g$. Elliptic Curves are Hyperelliptic Curves of genus 1, Hyperelliptic Curves exist from genus 1 to infinity. For hyperelliptic curves of genus $g \geq 2$, there is no natural group law on $C(K)$, you can't "add" points like you do on an elliptic curve as the points on a hyperelliptic curve don't form a group. A group law is defined via the $Jacobian\ Variety$ of C over a field, which is a finite abelian group.

The $Jacobian$ of the curve C is the quotient group $J = D^0/P$, where D^0 is the set of Divisors of degree zero, and P is the set of divisors of rational functions. The equivalence classes of the Jacobian are each represented by a unique $reduced$ divisor upon which we perform the group law. The Hyperelliptic Curve DLP is then: Let $\mathbb{F}_q$ be a finite field with q elements. Given two divisors D_1 and D_2 in the Jacobian, determine the integer $m \in \mathbb{Z}$, such that $D_2 = mD_1$.

Theorem 2.1 *Let J_C be the Jacobian variety of a hyperelliptic curve C. The group of $\mathbb{F}_q$ rational points on J_C is denoted by $J_C(\mathbb{F}_q)$. Let $\chi_q(t)$ be the characteristic polynomial of the q-th power Frobenius endomorphism of C. Then the order of the Jacobian is given by $\#J_C(\mathbb{F}_q) = \chi_q(1)$.*

The following inequality, which is known as the *Hasse-Weil* bound, bounds $\#J_C(F_q)$:

$$\lceil(\sqrt(q) - 1)^{2g}\rceil \leq \#J_C(\mathbb{F}_q) \leq \lfloor(\sqrt(q) + 1)^{2g}\rfloor$$

Definition 2.2 Consider a hyperelliptic curve of the form: $y^2 = f(x)$, where the degree of the $f(x)$ polynomial is 5 (ie. a genus 2 curve), and defined over a field $\mathbb{F}_q = \mathbb{F}_{p^l}$, where p is a prime number not equal to 2. The characteristic polynomial of the Frobenius endomorphism of C is then;

$$\chi_q(t) = t^4 - s_1t^3 + s_2t^2 - s_1qt + q^2, s_i \in \mathbb{Z},\ |s_1| \leq 4\sqrt(q),\ |s_2| \leq 6q$$

Due to Theorem 2.1 and Definition 2.2, the order of the Jacobian is then given by:

$$\#J_C(\mathbb{F}_q) = q^2 + 1 - s_1(q+1) + s_2,$$

$s_1 = 1+q-M_1$, $s_2 = (M_2-1-q^2+s_1^2)/2$, where M_i is the number of $\mathbb{F}_{q^i}$-rational points on the curve. This equation reduces the problem of finding the order of the Jacobian, to that of determining the number of points on the curve over the base field and quadratic extension field (for genus 2 curves). This method is impractical in general over large prime fields.

The following inequality provides a bound on s_2:

$$\lceil 2\sqrt(q)|s_1| - 2q\rceil \leq s_2 \leq \lfloor s_1^2/4 + 2q\rfloor$$

3 Point Counting methods for Fields of Characteristic 2

Here we detail the point counting methods for hyperelliptic curves over fields of characteristic 2 that have been implemented for this paper.

3.1 Koblitz method

Koblitz described a method [10] of calculating the number of points on the Jacobian of a hyperelliptic curve of genus 2 and of small characteristic, by using zeta functions.

Theorem 3.1 *Let C be a hyperelliptic curve of genus g defined over $\mathbb{F}_q$, and let $Z_C(t)$ be the zeta-function of C. Let $\mathbb{F}_{q^n}$ be the degree-n extension of $\mathbb{F}_q$, and let N_n denote the order of the finite abelian group $J_C(\mathbb{F}_{q^n})$. Denote by M_n the number of $\mathbb{F}_{q^n}$-rational points on C. Then we have the following theorem;*

(i)
$$Z_C(t) = \frac{P(t)}{(1-t)(1-qt)}$$

where $P(t)$ is a polynomial of degree 2g with integer coefficients. Moreover, $P(t)$ has the form:

$P(t) = 1 + a_1 t + ... + a_{g-1}t^{g-1} + a_g t^g + q a_{g-1} t^{g+1} + q^2 a_{g-2} t^{g+2} + ... + q^{g-1} a_1 t^{2g-1} + q^g t^{2g}$

(ii) *$P(t)$ factors as*

$$P(t) = \prod_{i=1}^{g} (1 - \alpha_i t)(1 - \bar{\alpha}_i t),$$

where each α_i is a complex number of absolute value $\sqrt{q}$ and $\bar{\alpha}_i$ denotes the complex conjugate of α_i.

(iii) *$N_n = \#J_C(\mathbb{F}_{q^n})$ satisfies*

$$N_n = \prod_{i=1}^{g} |1 - \alpha_i^n|^2,$$

where $||$ denotes the usual complex absolute value.

In order to compute N_n, it suffices to calculate the coefficients $a_1, a_2, ..., a_g$ of $P(t)$, factor $P(t)$ thus determining the α_i, and compute N_n by the above formula. An algorithm to determine the points on the Jacobian of a hyperelliptic curve of genus 2 is therefore;

Algorithm 1 *Koblitz method*
INPUT: A hyperelliptic curve C, of genus 2 over the field $\mathbb{F}_{q^n}$.
OUTPUT: $\#J_C(\mathbb{F}_{q^n})$, the order of the Jacobian of the field $\mathbb{F}_{q^n}$.

(i) *By exhaustive search, compute M_1 and M_2*

(ii) *The coefficients of $Z_C(t)$ are given by; $a_1 = M_1 - 1 - q$ and $a_2 = (M_2 - 1 - q^2 + a_1^2)/2$.*

(iii) *Solve the quadratic equation $X^2 + a_1 X + (a_2 - 2q) = 0$ to obtain two solutions λ_1 and λ_2.*

(iv) *Solve the quadratic equation $X^2 - \lambda_1 X + q = 0$ to obtain a solution α_1, and solve $X^2 - \lambda_2 X + q = 0$ to obtain a solution α_2.*

(v) *Then $\#J_C(\mathbb{F}_{q^n}) = |1 - \alpha_1^n|^2 \cdot |1 - \alpha_2^n|^2$.*

This method is only suitable for genus 2 curves, as it gets too difficult to calculate the coefficients of the Zeta function for $g > 2$. As one has to calculate M_1 and M_2, the number of points on the curve over $\mathbb{F}_q$ and $\mathbb{F}_{q^2}$, for genus 2 curves, the characteristic of the field should be small.

3.2 Sakai and Sakurai method

Sakai and Sakurai gave a point-counting method[11] for curves of small characteristic, but of arbitrary genus. Throughout this subsection, F denotes an algebraic function field of genus g whose constant field is the finite field $\mathbb{F}_q$ and P denotes the set of places of F/K.

Theorem 3.2 *Let the polynomial $L(t) = (1 - t)(1 - qt)Z(t)$ be called the $L - polynomial$ of the function field $F/\mathbb{F}_q$, where $Z(t)$ denotes the zeta function of*

$F/\mathbb{F}_q$. Then the following holds;

(i) $L(t) \in Z[t]$ *and deg* $L(t) = 2g$.

(ii) $L(t) = q^g t^{2g} L(1/qt)$

(iii) $L(1) = h$*, the class number of* $F/\mathbb{F}_q$.

(iv) *We write* $L(t) = \sum_{i=0}^{2g} a_i t^i$. *Then the following holds;*
   (a) $a_0 = 1$ *and* $a_{2g} = q^g$
   (b) $a_{2g-i} = q^{g-i} a_i$ *for* $0 \leq i \leq g$.
   (c) $a_1 = N - (q+1)$ *where N is the number of places* $P \in P_F$ *of degree one.*

(v) $L(t)$ *factors in* $C[t]$ *in the form* $L(t) = \prod_{i=1}^{2g}(1 - \alpha_i t)$. *The complex numbers* $\alpha_1, ..., \alpha_{2g}$ *are algebraic integers, and they can be arranged in such a way that* $\alpha_i \alpha_{g+i} = q$ *holds for* $i = 1, ..., g$.

(vi) *If* $L_r(t) = (1-t)(1-q^r t)Z_r(t)$ *denotes the L-polynomial of the constant field expression* $F_r = F\mathbb{F}_{q^r}$, *then* $L_r(t) = \prod_{i=1}^{2g}(1 - \alpha_i^r t)$.

The order of the Jacobian of curves of small characteristic and of arbitrary genus can thus be determined by the following algorithm;

Algorithm 2 *Sakai and Sakurai method*

```
INPUT: Hyperelliptic Curve $C : v^2 + h(u)v = f(u)$ over $F_{q^n}$
OUTPUT: $\#J_C(\mathbb{F}_{q^n})$
(i) Determine $N_r$, the number of points on the curve over $\mathbb{F}_{q^r}$ for $r = 1, ..., g$.
(ii) Determine the coefficients of $L_{\mathbb{F}_q}(t) = \sum_{i=0}^{2g} a_i t^i$ in the following way;
    (a) $a_0 = 1$
    (b) for $1 \leq i \leq g : a_i = (\sum_{k=1}^{i}(N_k - (q^k + 1))a_{i-k})/i$.
    (c) for $g + 1 \leq i \leq 2g : a_i = q^{i-g} a_{2g-i}$.
(iii) Compute $L_{\mathbb{F}_{q^n}}(1) = \prod_{k=1}^{n} L_{\mathbb{F}_q}(\zeta^k)$ , where $\zeta$ runs over the $n$-th root of unity
(iv) Return $\#J_C(\mathbb{F}_{q^n}) = L_{\mathbb{F}_{q^n}}(1)$.
```

Note that it should be easy to count $N_1, ..., N_g$ if $\mathbb{F}_q$ is small, so this algorithm is only suitable for fields of small characteristic.

3.3 Timings for Characteristic 2 Methods

Here we present timings for the curve $C_1 : v^2 + v = u^5 + u^3 + u$, over $\mathbb{F}_{2^{101}}$ and $C_2 : v^2 + (u^2 + u + 1)v = u^5 + u + 1$ over $F_{2^{89}}$

Curve	Koblitz Method (s)	Sakai Method (s)
C_1	0.057	26.335
C_2	0.074	23.790

Clearly the Koblitz method is far superior to the Sakai and Sakurai method for the two sample curves. The Koblitz method has the disadvantage though that it only works for genus 2 curves, whereas the Sakai and Sakurai method extends easily to arbitrary genus curves.

To illustrate the importance of determining the group order for the security of the cryptosystem, consider curve C_1 above. This curve is *not* secure, as the order of the Jacobian is:
$\#J_{C_1}(F_{2^{101}}) = 7{*}607{*}1512768222413735255864403005264105839324374778520631 853993$

The largest prime factor of $\#J_{(C_1)}(F_{2^{101}})$ divides $(2^{101})^3 - 1$, which makes it vulnerable to the Frey-Rück attack[20].

4 Point Counting methods for Prime Fields

Here we detail the point counting methods for hyperelliptic curves over prime fields, that have been implemented for this paper.

4.1 Hasse-Witt method

There is a method[12] to calculate the order of the Jacobian over a prime field using the Hasse-Witt matrix[14], also known as the Cartier-Manin operator. The following method only works for prime fields, not prime extension fields, and only for a cryptographically-insecure prime (the algorithm is infeasible when p is greater than around 100000).

Theorem 4.1 ***Theorem.*** *Let $y^2 = f(x)$ with $deg\ f = 2g + 1$ be the equation of a genus g hyperelliptic curve. Let c_i be the coefficient of x^i in the polynomial $f(x)^{(p-1)/2}$. Then the Hasse-Witt matrix is given by $A = (c_{ip-j})_{1 \leq i,j \leq g}$.*

Lemma 4.2 *Let c_i be the coefficient of x^i in $f(x)^{(p-1)/2}$ as detailed in the above theorem. Therefore, s_i in the formula for $\#J_C(\mathbb{F}_q)$ given in section 2 is; $s_1 \equiv c_{p-1} + c_{2p-2} \pmod p$ and $s_2 \equiv c_{p-1}c_{2p-2} + c_{p-2}c_{2p-1} \pmod p$.*

Since $|s_1| \leq 4\sqrt(p)$, if $p > 64$ then s_1 is uniquely determined by the formula in the above lemma. Also, there are at most 5 possibilities for s_2, due to the bound given in section 2. The algorithm is defined as follows;

Algorithm 3 *Hasse-Witt Method*
INPUT: A polynomial $f(x)$ defined over $\mathbb{F}_p$, p a prime, $64 < p < 100000$, and $deg f(x) = 5$.
OUTPUT: $\#J_C(\mathbb{F}_p)$

```
(i) Raise f(x) to the power of $(p - 1)/2$.
(ii) Extract the 4 coefficients $c_{ip-j}, 1 \leq i, j \leq g$
(iii) Put $s_1 = c_{p-1} + c_{2p-2} \pmod p$
(iv) Determine the 5 possible values S for $s_2$;
     $s_2 \leftarrow c_{p-1}c_{2p-2} + c_{p-2}c_{2p-1}\ (mod\ p)$.
     if $s_2$ even: S $\leftarrow \{s_2 + 2mp \mid 2\sqrt{p}|s_1| - 2p \leq s_2 + 2mp \leq s_1^2/4 + 2p\}$,
     else S $\leftarrow \{s_2 + (2m+1)p \mid 2\sqrt{p}|s_1| - 2p \leq s_2 + (2m+1)p \leq s_1^2/4 + 2p\}$.
```

(v) *Determine the list L of candidates for* $\#J_C(\mathbb{F}_p)$*;*
$L \leftarrow \{1 + p^2 - s_1(p + 1) + s_2 | \; s_2 \in S\}$.

(vi) *If* $\#L = 1$*, then return the unique element of L,*
else determine $\#J_C(\mathbb{F}_p)$ *by multiplying a random point D on* $J_C(\mathbb{F}_q)$ *by each element of L.*

It is difficult to calculate $s_i \pmod p$ by this method, even when $q = p$, and $g = 2$, which is why it is only suitable for small p. Here is a set of timings for the curve $C_1 : v^2 = u^5 + 3u$ over the prime field $\mathbb{F}_p$

Prime Field $\mathbb{F}_p$	$\#J_C(\mathbb{F}_p)$	Time (s)
10433	110699362	1.822
20201	414268018	4.125
40577	1657567138	8.756

This method is too slow and consumes too many system resources! Notice that the time to compute the order of the Jacobian roughly doubles when the prime field doubles, which will lead to unacceptably long times as the prime reaches cryptographic size. Seeing as the f polynomial of the curve is being raised to the power of $(p - 1)/2$, this method will take up a very large amount of memory and is infeasible for a large p.

4.2 Furukawa, Kawazoe, Takahashi method

This is a point-counting method [12] for curves of type $C : y^2 = x^5 + ax$ (hence the curve is of genus 2), over a prime field $\mathbb{F}_p$. p must be greater than 64 and congruent to $1 \bmod 8$. The jacobi symbol $(a|p)$ must also be equal to -1. Two variants of this method exist, a recent paper [13] removed some of the steps in the algorithm, and replaced them with an explicit formula. When the preconditions of these algorithms are met, there is a fast way to evaluate s_1 and s_2 without having to calculate $f(x)^{(p-1)/2}$ as in the Hasse-Witt method.

Theorem 4.3 *Let* a *be an element of* $\mathbb{F}_p$*,* C *a hyperelliptic curve defined by the equation* $y^2 = x^5 + ax$ *and* $\chi_p(t)$ *the characteristic polynomial of the* p*-th power Frobenius endomorphism of C. If* $p \equiv 1 \pmod 8$*, then* s_1*,* s_2 *in* $\chi_p(t)$ *are given as follows;*

$$s_1 \equiv (-1)^{(p-1)/8} 2c(a^{3(p-1)/8} + a^{(p-1)/8}) \pmod p$$
$$s_2 \equiv 4c^2 a^{(p-1)/2} \pmod p$$

where c is an integer such that $p = c^2 + 2d^2$*,* $c \equiv 1 \pmod 4$ *and* $d \in \mathbb{Z}$*.*

An algorithm to count points on the Jacobian of a curve of this special type is as follows;

Algorithm 4 *Furukawa Method (1)*

INPUT: $a \in F_p$*, where* $p \equiv 1 \pmod 8$ *and* $p > 64$
OUTPUT: $\#J_C(\mathbb{F}_p)$ *(C: a hyperelliptic curve of genus 2 defined by* $y^2 = x^5 + ax$*).*

(i) *Calculate an integer c such that* $p = c^2 + 2d^2$, $c \equiv 1 \pmod 4$, $d \in \mathbb{Z}$ *by using Cornacchia's algorithm.*

(ii) *Determine* s_1*:*
$s \leftarrow (-1)^{(p-1)/8} 2c(a^{3(p-1)/8} + a^{(p-1)/8}) \pmod p \qquad (0 \le s \le p-1)$
If $s < 4\sqrt{(p)}$*, then* $s_1 \leftarrow s$*, else* $s_1 \leftarrow s - p$*.*

(iii) *Determine the list* S *of candidates for* s_2*;*
$t \leftarrow 4c^2 a^{(p-1)/2}$ *(mod p)* $\qquad (0 \le t \le (p-1)$
if t even: $S \leftarrow \{t + 2mp \mid 2\sqrt{p}|s_1| - 2p \le t + 2mp \le s_1^2/4 + 2p\}$,
else $S \leftarrow \{t + (2m+1)p \mid 2\sqrt{p}|s_1| - 2p \le t + (2m+1)p \le s_1^2/4 + 2p\}$.

(iv) *Determine the list L of candidates for* $\#J_C(\mathbb{F}_p)$*;*
$L \leftarrow \{1 + p^2 - s_1(p+1) + s_2 | s_2 \in S\}$.

(v) *If* $\#L = 1$*, then return the unique element of L,*
else determine $\#J_C(\mathbb{F}_p)$ *by multiplying a random point* D *on* $J_C(\mathbb{F}_q)$ *by each element of* L*.*

The second variant does not need to calculate s_1 and s_2. Integers c and d are calculated using Cornacchia's algorithm as above, with the extra condition that $2d \equiv -(a^f + a^{3f})c \pmod p$, where $f = (p-1)/8$. The characteristic polynomial of the p-th power Frobenius map for C is then given by;

$$\chi(t) = t^4 + (-1)^f 4dt^3 + 8d^2t^2 + (-1)^f 4dpt + p^2$$

and therefore;

$$|J_C(\mathbb{F}_p)| = 1 + (-1)^f 4d + 8d^2 + (-1)^f 4dp + p^2$$

Here are a set of timings for the curve $C_1 : y^2 = x^5 + 3x$ over the field $\mathbb{F}_{1208925819614629175095961}$;

Curve	Furukawa 1 (s)	Furukawa 2 (s)
C_1	0.034	0.031

As can be seen, both variants of this method are extremely quick, but are very specific in terms of the curve and the prime field. This possible disadvantage will be discussed later on in this paper.

4.3 Koblitz method

Koblitz gave a solution [10] for curves of the form $y^2 + y = x^n$, $n = 2g+1$, $p \equiv 1 \bmod n$, that relies on evaluating jacobi sums. Let $\alpha \in \mathbb{F}_p$ be a fixed non-n-th power (as we're dealing with genus 2 curves, α will be a fixed non-quintic-power). There is a unique multiplicative map χ on $\mathbb{F}_p^*$ such that $\chi(\alpha) = \zeta$.

Definition 4.4 Jacobi Sum: $J_r(\chi, \chi) = \sum_{t \in \mathbb{F}_{p^r}} \chi(t)\chi(1-t)$

Then $\#J_C(\mathbb{F}_p) = \prod_{i=1}^{n-1} \sigma_i(J(\chi,\chi) + 1)$, where σ is an automorphism of the field $\mathbb{Q}(\zeta)$ such that $\sigma_i(\zeta) = \zeta^i$. Evaluating the jacobi sum is very inefficient using the formula given above, as it requires iterating through every element in the field, which is obviously impractical for a large prime field.

4.3.1 Mersenne Primes

There is a quick way [10] of evaluating the jacobi sum if the prime p is a generalised Mersenne Prime, ie. of the form $p = \frac{a^n-1}{a-1}$. If α, the non-n-th power mentioned earlier is chosen such that $\alpha^{(p-1)/n} \equiv a \pmod{p}$; the Jacobi Sum can then be evaluated as;

$$J(\chi,\chi) = \pm\zeta^k \prod_{i=1}^{g} (a - \sigma_i^{-1}(\zeta))$$

$\pm\zeta^k$ is chosen such that $J(\chi,\chi) \equiv -1 \pmod{(\zeta-1)^2}$ in the ring $\mathbb{Z}[\zeta]$ [17]. $\pm\zeta^k$ is uniquely determined by the value of $a \bmod n$. In the case $n = 5$, this root of unity is given by;

a mod 5	0	2	3	4
$\pm\zeta^k$	$-\zeta$	$-\zeta^4$	ζ^2	ζ^3

Here are timings for the curve $C_1 : v^2 + v = \alpha u^5$, which is a twist of the curve $v^2 + v = u^5$, over $\mathbb{F}_p$, using the Mersenne Prime Method;

Prime Field $\mathbb{F}_p$	a	Time (s)
100013000640014200121	100003	0.357

Clearly this method is very efficient. The order of the Jacobian of the above curve and field is;

$$\#J_C(\mathbb{F}_p) = 5 * 2000520059203862158324190070180683302981$$

4.3.2 LLL algorithm

If p is not a generalised Mersenne Prime, it is still possible[18] to evaluate the jacobi sum by using the LLL[16] algorithm. LLL is used to find a short vector in the lattice in $\mathbb{R}^{n-1}$ corresponding to the ideal generated over $\mathbb{Z}[\zeta]$ by the two elements p and $(\zeta - a)$, where the integer a is an n-th root of unity modulo p described earlier. We leave out the theory in the following section for briefness, for a more complete description of the algorithm see [18]. The prime ideal P has $\mathbb{Z}$-basis;

$$\{p, \zeta - a, \zeta^2 - a_2, ..., \zeta^{n-2} - a_{n-2}\}$$

where $a_k = (a^k \bmod p)$. The Gram matrix with respect to this basis is given by;

$< p, p >$	$(n-1)p^2$
$< p, \zeta^k - a_k >$	$-p - (n-1)pa_k$
$< \zeta^k - a_k, \zeta^l - a_l >$	$\delta_{kl} n - 1 + a_k + a_l + (n-1)a_k a_l$

where $\delta_{kl} = 1$, if $k = l$, and $\delta_{kl} = 0$ otherwise. Seeing as all the entries of the Gram matrix are integral, we need to pass the Gram matrix through as input to the integral LLL-algorithm, and a transformation matrix will be obtained as output. To obtain the generator β of P, we multiply the transformation matrix that LLL outputs, with the $\mathbb{Z}$-basis defined earlier.

We then set; $\tilde{J} = \prod_{i=1}^{g} \sigma_i^{-1}(\beta)$. $\tilde{J}$ is equal to $J(\chi, \chi)$ up to a root of unity, ie. $J(\chi, \chi) = r\zeta^s \tilde{J} \equiv -1 \pmod{(\zeta - 1)^2}$. This congruence can be easily solved by finding $r \in \{\pm 1\}$ such that $r \equiv -\sum a_j (\bmod n)$, and then finding $s \equiv r \sum j a_j (\bmod n)$.

An algorithm to find the order of the Jacobian for curves of this type is as follows;

Algorithm 5 *Koblitz Method*
INPUT: $C : v^2 + v = u^5$ *over* $\mathbb{F}_p$*, where* $p \equiv 1 \pmod 5$
OUTPUT: $\#J_C(\mathbb{F}_p)$

(i) *Find an integer a such that* $\alpha^{(p-1)/5} \equiv a \pmod p$*, where* α *is a fixed non-quintic-power.*

(ii) *Evaluate the* $\mathbb{Z}$*-basis* $b = (b_0, b_1, b_2, b_3)$.

(iii) *Construct a gram matrix G for the* $\mathbb{Z}$*-basis* b.

(iv) *Get a transformation matrix* $H = (h_{ij})$ *from the integral LLL algorithm for G*

(v) *Find the generator of the prime ideal,* $\beta = \sum_{i=0}^{3} b_i h_{0i}$

(vi) *Evaluate* $\tilde{J} = \prod_{i=1}^{g} \sigma_i^{-1}(\beta)$

(vii) *Evaluate* $J(\chi, \chi) = r\zeta^s \tilde{J}$

(viii) *Evaluate* $\#J_C(\mathbb{F}_p) = \prod_{i=1}^{n-1} \sigma_i(J(\chi, \chi) + 1)$

Here are timings for the curve $C_1 : v^2 + v = u^5$ over $\mathbb{F}_p$;

Prime Field $\mathbb{F}_p$	a	Time (s)
100013000640014200121	10000600009	0.3636
100013000640014200121	1000090002700027	0.3727

Despite having to find a reduced basis using the LLL algorithm, this algorithm compares well with the previous algorithm. Both the above methods can also calculate *twists* of the curve by non-n-th powers and by non-squares.

5 Commentary

As remarked earlier, Gaudry's point-counting algorithm [8] runs in about a week for a genus 2 curve defined over an 80-bit field. Most of the algorithms detailed in this paper are significantly faster than this; a researcher trying to determine the security of a curve over a particular field might not want to wait a week to find out.

On the other hand, the efficient prime-field algorithms detailed in this paper are constrained to particular types of curves. It can be a bad idea in terms of security to focus in on one particular class of curve. For example, it is now known that no secure curve exists with genus 2 among those defined over $\mathbb{F}_2$ and $h(u) = 1$.

There is little difference between the running times of the Prime Field and Fields of Characteristic 2 algorithms, both are fast. The characteristic 2 algorithms though are restricted to curves defined over $\mathbb{F}_2$, not those over $\mathbb{F}_{2^n}$. New algorithms [19] can handle curves over $\mathbb{F}_{2^n}$ efficiently, there is a lot of ongoing research in this area. However, many curves exist defined over $\mathbb{F}_2$ which have no known weakness.

As commented earlier, increased field size equates to increased security. Improved attacks are an ongoing reality, leading to ever increasing field sizes being used. For example, Gaudry recently presented an attack [9] on genus 3 curves, which means key sizes must be increased by 12% to maintain security levels. It is therefore important when assessing point-counting algorithms, that scalability in terms of field size is taken into account. The algorithms presented in this paper scale easily over bigger fields, with the obvious exception of the Hasse-Witt matrix method.

6 Addendum

All timings were conducted on an 800mhz Duron processor with 756 megabytes of ram, running Debian GNU/Linux, kernel 2.4.26. The code was written in C++, using the MIRACL [15] library. The code used for this paper is released under the GNU General Public License, and is available at;
http://www.computing.dcu.ie/˜coheigeartaigh/crypto.html

References

[1] Adleman, L., *A subexponential algorithm for the discrete logarithm problem with applications to cryptography*, Proc. 20th IEEE Found. Comp. Sci. Symp., 1979, 55-60.

[2] Rivest, R.L., A. Shamir and L. Adlemann, *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, Communications of the ACM. **21**, Nr.2 (1978), 120-126

[3] ElGamal, T., *A Public Key Cryptosystem and a Signature Scheme based on Discrete Logarithms*, IEEE Transactions on Information Theory, **31**, Nr.4 (1985), S.469-472

[4] Menezes, A., "Elliptic Curve Cryptosystems", CryptoBytes, **1**, No.2 (1995).

[5] Menzes, A., M. Jacobson and A. Stein, "Hyperelliptic curves and cryptography", Fields Institute Communications Series, **41** (2004), 255-282.

[6] Boston, N., T. Clancy, Y. Liow, and J. Webster, *Genus Two Hyperelliptic Curve Coprocessor*, Workshop on Cryptographic Hardware and Embedded Systems — CHES 2002, Springer Verlag, LNCS 2523, New York (2002).

[7] Pohlig, S., and M. Hellman, *An improved algorithm for computing discrete logarithms over GF(p) and its cryptographic significance*, IEEE Transactions on Information Theory, **24** (1978), 106-110

[8] Gaudry, P., and R. Harley, *Counting points on hyperelliptic curves over finite fields*, ANTS-IV, Springer-Verlag, LNCS **1838** (2000), 313-332.

[9] Gaudry, P., and E. Thomé, *A double large prime variation for small genus hyperelliptic index calculus*, Cryptology ePrint archive;
http://eprint.iacr.org/2004/153.pdf

[10] Koblitz, N., "Algebraic Aspects of Cryptography", Springer Verlag, Algorithms and Computation in Mathematics, **3** (1999)

[11] Sakai, Y., and K. Sakurai, *On the Practical Performance of Hyperelliptic Curve Cryptosystems in Software Implementation*, IECE Trans. Fundamentals, vol. E83-A, No. **4**, April 2000.

[12] Furukawa, E., M. Kawazoe, and T. Takahashi, *Counting Points for Hyperelliptic Curves over Finite Prime Fields*, Cryptology ePrint archive;
http://eprint.iacr.org/2002/181.pdf

[13] Haneda, M., M. Kawazoe, and T. Takahashi, *Suitable Curves for Genus-4 HCC over Prime Fields Point Counting Formulae for Hyperelliptic Curves of type* $y^2 = x^{2k+1} + ax$, Cryptology ePrint archive;
http://eprint.iacr.org/2004/151.pdf

[14] Manin, Ju. I., *The Hasse-Witt matrix of an Algebraic Curve*, Ameri. Math. Soc., Transl. Ser. **45** (1965), 245-264.

[15] MIRACL (Multiprecision Integer and Rational Arithmetic C/C++ Library)
http://indigo.ie/˜mscott/

[16] Lenstra, A.K., H.W. Lenstra Jr., and L. Lovász, *Factoring polynomials with rational coefficients*, Math. Ann., **261**, 515-534.

[17] Ireland, K., and M.I. Rosen, *A Classical Introduction to Modern Number Theory*, 2nd edition (1990), Springer Verlag, 227.

[18] Buhler, J., and N. Koblitz, *Lattice basis reduction: Jacobi sums and hyperelliptic cryptosystems*, Bull. Austral. Math. Soc., **57**, 147-154.

[19] Vercauteren, F., *Computing zeta functions of hyperelliptic curves over finite fields of characteristic 2*, in Moti Young (Ed.) Advances in cryptology - CRYPTO 2002, Lecture Notes in Computer Science **2442**, Springer 2002, 369-384.

[20] Frey, G., and H. Rück, *A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves*, Math. Comp. **62**, 865-874.